Movie Mail Final Project By Samuel Bartolome

Table of Content

Movie Mail Website Application:

Problem Description……………………………………………………………………………………………………2 – 3.

Problem Statement……………………………………………………………………………………………………4.

Context Diagram………………………………………………………………………………………………………5.

Use Case Diagram……………………………………………………………………………………………………6.

User Story Map……7.

User Stories………8.

Acceptance Criteria…………………………………………………………………………………………………9.

Order Use Scenarios…………………………………………………………………………………………………10.

Activity Diagrams……………………………………………………………………………………………………11-12.

State Machine Diagrams……………………………………………………………………………………………13.

Architecture Diagram………………………………………………………………………………………………14.

Class Diagram……15.

ER Movie Mail Data Base…………………………………………………………………………………………16.

ER Movie Mail Login Data Base…………………………………………………………………………………17.

Sequence Diagram……………………………………………………………………………………………………18-19.

User Interface……20-22.

END Points RESTFull API Project………………………………………………………………………………23-24.

Conclusion………25.

Problem Description:

Moviemail: Online DVD rental

We need to specify the requirements of a new system called MovieMail that allows customers to order DVD's online and receive and return DVD's by regular mail. This way you do not have to go to the physical video rental store.

Customers can order a subscription at MovieMail. The following subscriptions are offered:

Name subscription	Number of DVD's you may have at home	Number of DVD's per month	Price per month
Platina	4	6	$15,00
Gold	3	5	$13,00
Silver	2	4	$11,00
Bronze	1	3	$9,00
Basic	1	2	$7,00

Customers can subscribe online and pay in two different ways:

a. creditcard
b. bank transfer

If the customers pays with creditcard, the system will use an external system called Epay.com. Epay.com is a system that can be called through webservices and offers the following services:

1. validate creditcards
2. handle the creditcard payment

If Epay.com is not available, then we don't accept subscriptions that are paid with creditcard.

If the customer wants to pays with a bank transfer, the system will collect the bank information from the customer, and an MovieMail employee will perform the actual bank transfer with the online bank application of their local bank. If the bank transfer succeeded, the employee will record this in the MovieMail system.

After successfully ordering a subscription, the customer will receive the account login information by email.

Customers can have more than 1 subscriptions. Every subscription has its own account with their own username/password combination.

When the customer logs into the system, he/she can place DVD's on a watchlist. You are allowed to place a maximum of 50 DVD's on the watchlist. Customers can change the watchlist at any time. The watchlist has a certain order. The higher the order of the DVD on the watchlist, the higher the change that you will receive this DVD. If the customer receives a DVD, he/she can keep this DVD as long as you want at home.

If a customer wants to see a DVD that is not in the catalog of MovieMail, then you can place a request to add this DVD into the DVD catalog.

Every DVD has a unique scan code. If a DVD is send out, or received back, the DVD is scanned so that the system knows exactly where the DVD's are at the moment. The system prints the return envelopes automatically.

The system can show for every customer which DVD's he/she has at home, or has returned to MovieMail.

The system offers the following reporting functionality:

1. The top x (x is a variable) most popular DVD's
2. The top x most requested DVD's
3. An overview of all subscriptions and how much we earned.
4. The top x least popular DVD's

At MovieMail we have a CRM system where we store all relevant information from customers. When a customer orders a movie subscription, we want to add the customer information in our CRM system so we can use that information for advertising purposes.

It is not possible to end a subscription before the ending date, and it is also not possible to get money back from your ordered subscription.

Problem Statement:

For this project we are trying to bring a full functionally WebSite Application for customers who wants to rent movies online without the need of going to the store in person. For this main purpose the website offers the following features:

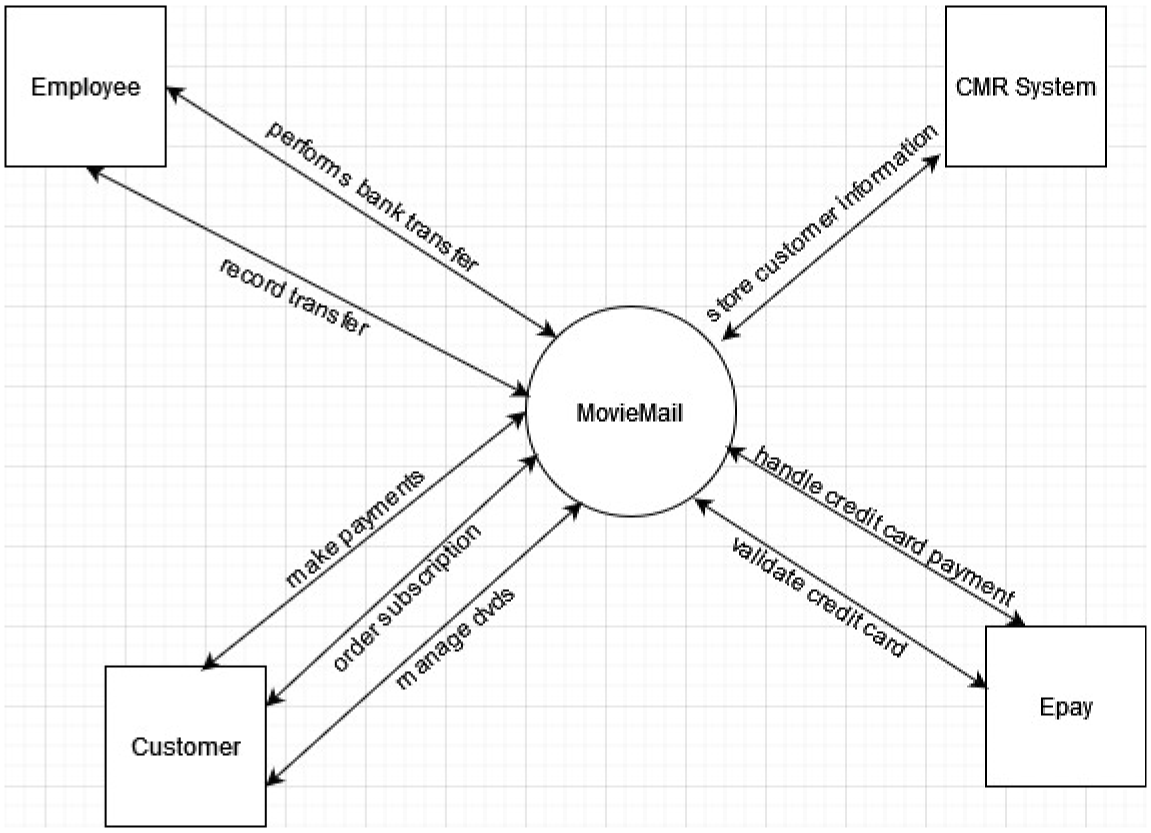
- The MovieMail System can do the following features:
  - The MovieMail system collects bank information from the customer.
  - It prints the return envelopes automatically
  - It reports most X popular DVDs
  - It reports most X requested DVDs
  - It overviews all subscriptions and how much the business has earned
  - It adds the customer information in our CMR system
- On the other hand, the customer can use the platform using these features:
  - He can order (rent) DVDs
  - He can receive DVDs
  - He can return DVDs
  - He can order (buy) a subscription
  - He can pay using credit card
  - He can pay using bank transfer
  - He can receive login account information
  - He can have more than 1 subscription
  - He can place DVDs on a watchlist (maximum of 50 DVDs)
  - He can manage his watchlist
  - He can place a request of a new DVD in the DVD catalog
  - He can receive the account login information by email
- Works with external entities who let the business to validate payments through credit card or bank accounts as well as store the customer information, handle it by an employee
- For credit card the payment is automatically validated and handle it by an external entity called Epay.

Agile methodologies combining with scrum requirements.

- The idea for this project is simple. Make the main use cases, create the appropriate story map for each of them and then create small user story cases which will be accepted for the product owner and develop, implement, and tested by the scrum team on each sprint, applying the best practices from these methodologies. Unfortunately, in this case I am the Product Owner, Scrum Master and Scrum Team so I will try to do my best to present the design and analysis in the following pages through the different diagrams I have created during course with the necessary modifications for this final project presentation.

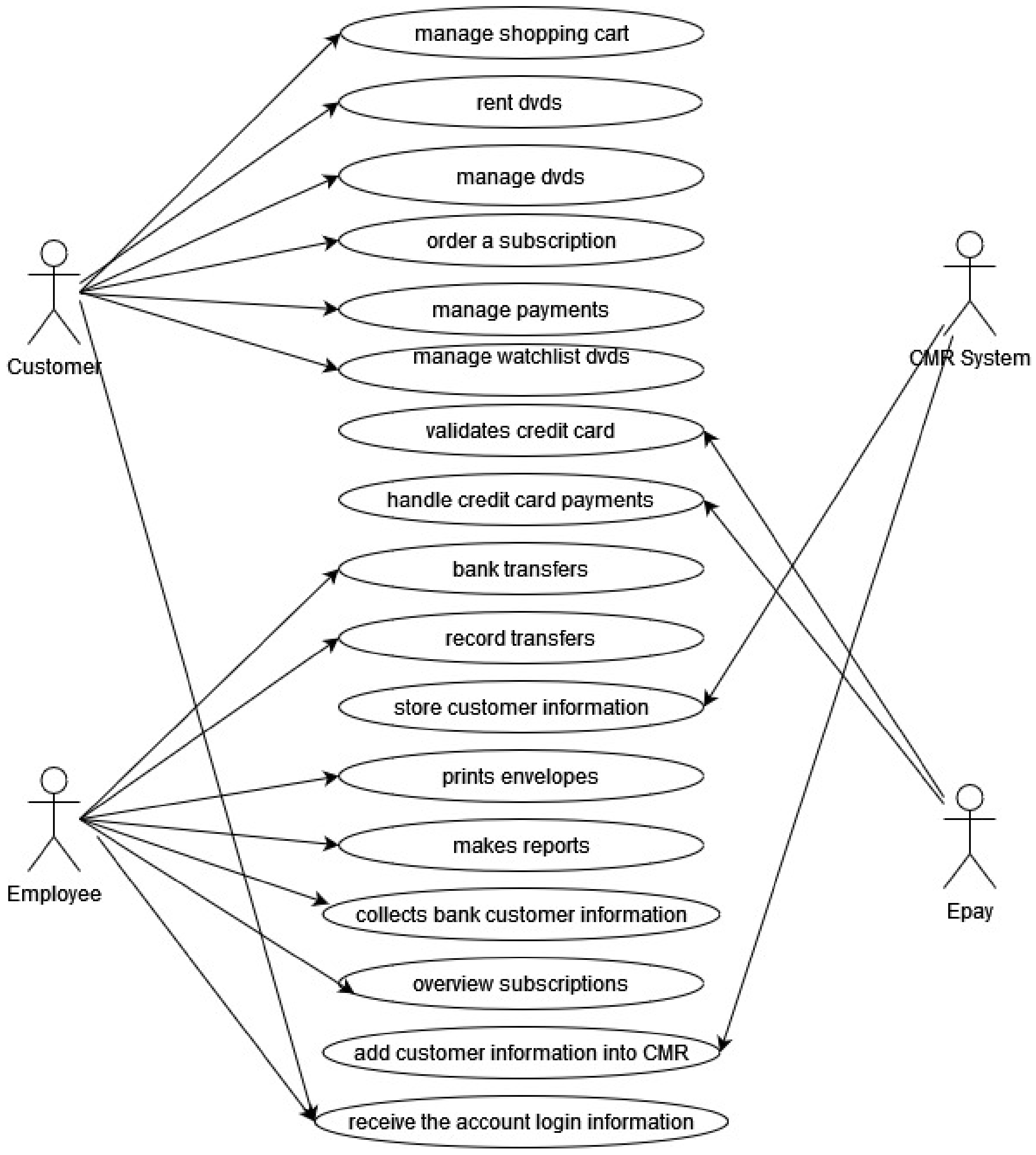
Context Diagram:

This context diagram brings the main features and external entities which interact with the system. Even though I have not been able to implement all of them properly in my project, I think it is worth it and important to leave the original diagram I created for this purpose.



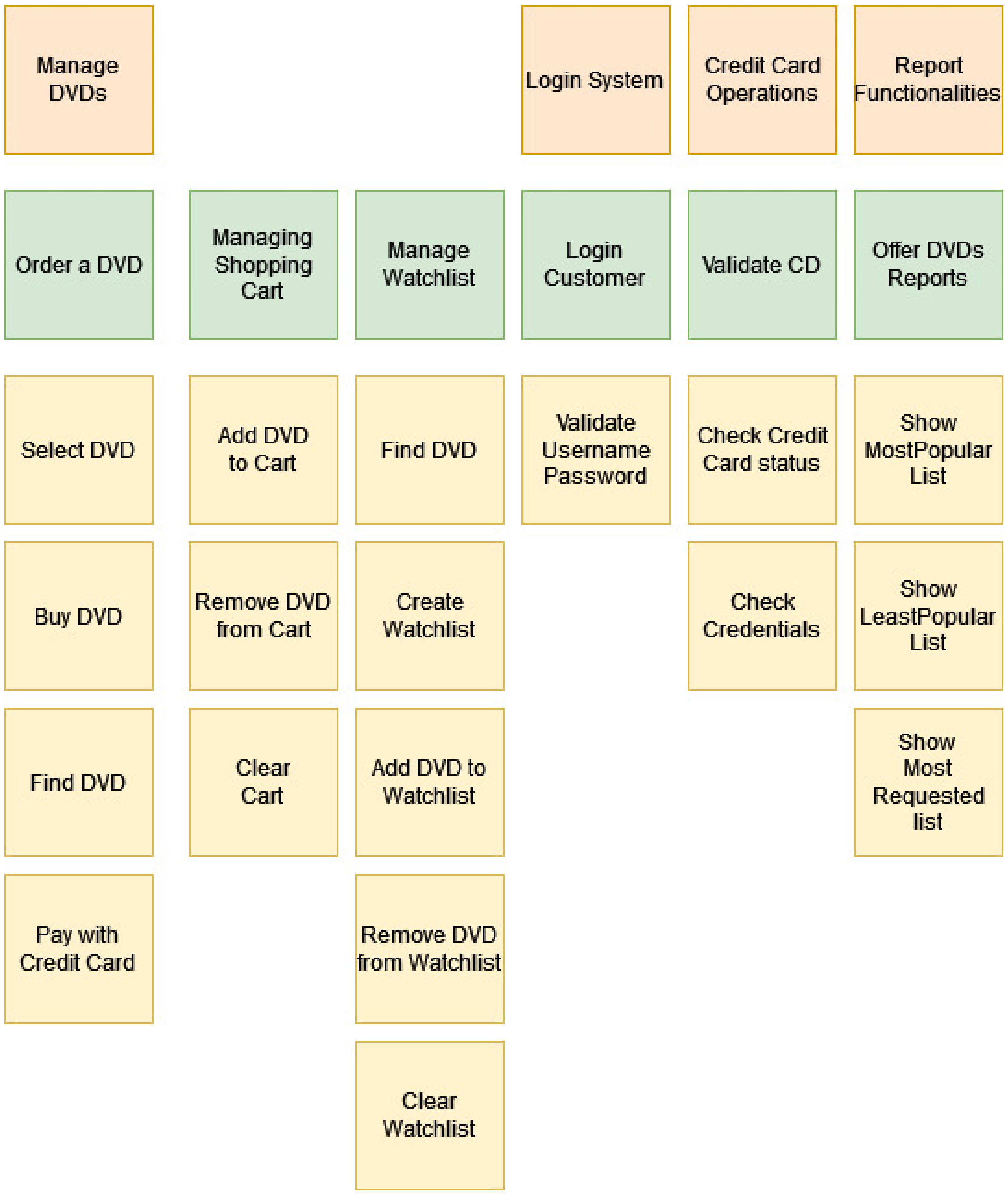
Use Case Diagram:

In my main use case diagram, I have brought the main features that this project needs to be implemented taking in account the main actors, especially the Customer and the Employee. For my project I have been able to implement many of the customer features but not all of them. I could not implement the employee's one. The Epay's system is implemented in a basic way where it validates the credit card but without checking important testing like the balance of the credit card for example.



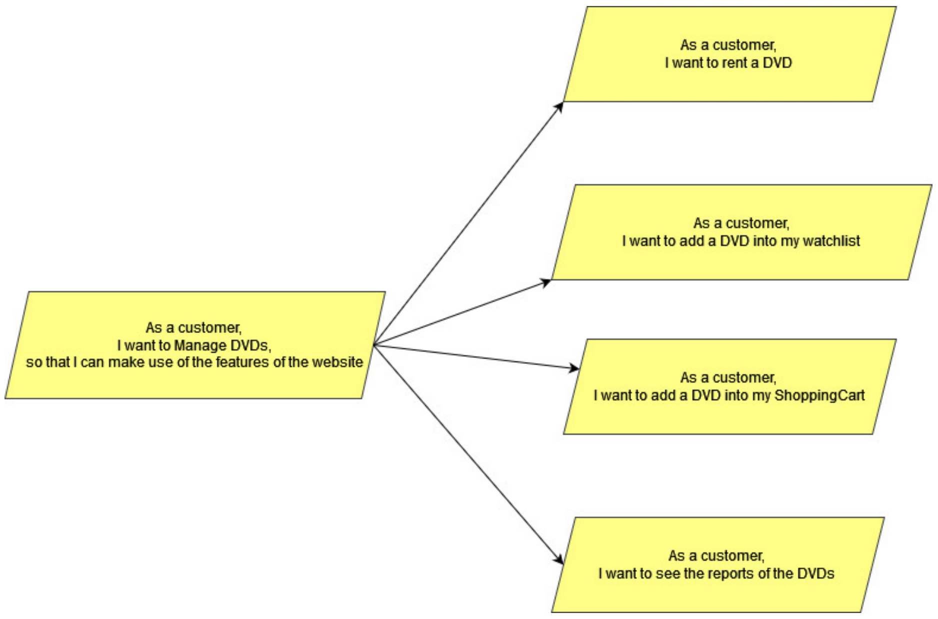
User Story Map:

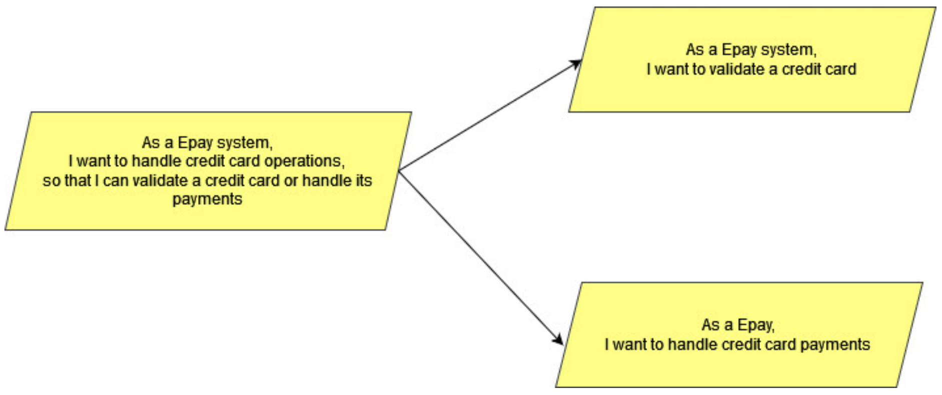
In this story map of my project contains the main user stories that I have been able to implement and make them functional (but not perfectly functional without testing possible bugs etc.) and works properly in the project. They are dividing in small user stories when I have implemented for most of them and end point in my RESTFull API project.



User Stories:

Here I bring two of the main user stories I have been able to implement in my project:


As a customer,
I want to Manage DVDs,
so that I can make use of the features of the website
As a customer,
I want to rent a DVD
As a customer,
I want to add a DVD into my watchlist
As a customer,
I want to add a DVD into my ShoppingCart
As a customer,
I want to see the reports of the DVDs


As a Epay system,
I want to handle credit card operations,
so that I can validate a credit card or handle its payments
As a Epay system,
I want to validate a credit card
As a Epay,
I want to handle credit card payments

Acceptance Criteria:

Acceptance criteria of the main features of the implemented project:

As a customer,
I want to Order a DVD,
so that I can select the one I like and watch it at home

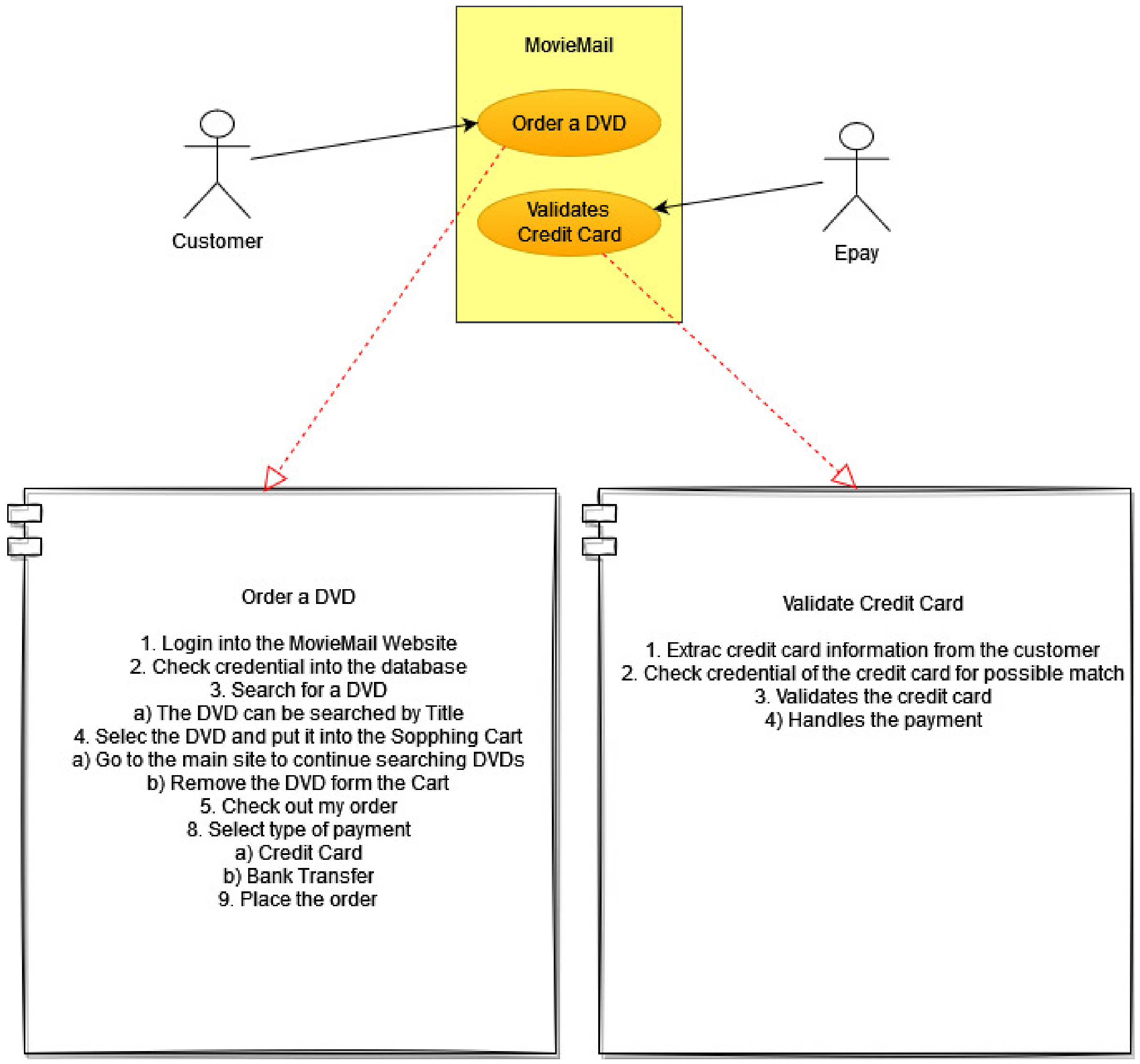
Login in the MovieMail Website
Find the DVD I want to order
Select the DVD an put it into my Shopping cart
add more DVDs or Check out my Shopping cart
Select my type of payment
Place the order

As a customer,
I want to manage the different features the web offers for its DVDs
so that I can create my own watchlist, request DVDs etc.

Login in the MovieMail Website
Find a DVD
Select the DVD by Name
Add or remove a DVD to my Watchlist
Clear my Watchlist
Add or remove a DVD into my Shopping Cart
Request List of MostPopular DVDs
Request List of Least Popular DVDs
Request List of Most Requested DVDs

Order Use Scenarios:

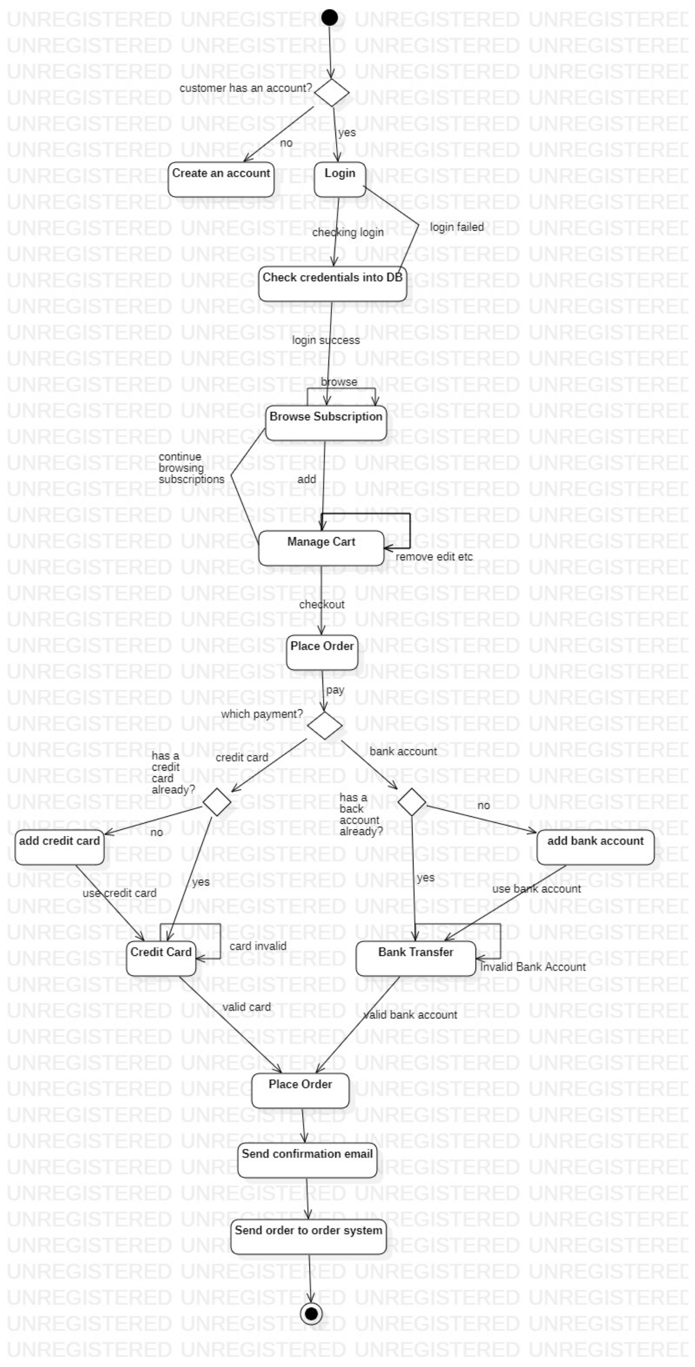
Two of the main order scenarios which are implemented and working on the RESTFull API project.


MovieMail
Order a DVD
Validates Credit Card
Customer
Epay
Order a DVD
1. Login into the MovieMail Website
2. Check credential into the database
3. Search for a DVD
a) The DVD can be searched by Title
4. Selec the DVD and put it into the Sopphing Cart
a) Go to the main site to continue searching DVDs
b) Remove the DVD form the Cart
5. Check out my order
8. Select type of payment
a) Credit Card
b) Bank Transfer
9. Place the order
Validate Credit Card
1. Extrac credit card information from the customer
2. Check credential of the credit card for possible match
3. Validates the credit card
4) Handles the payment

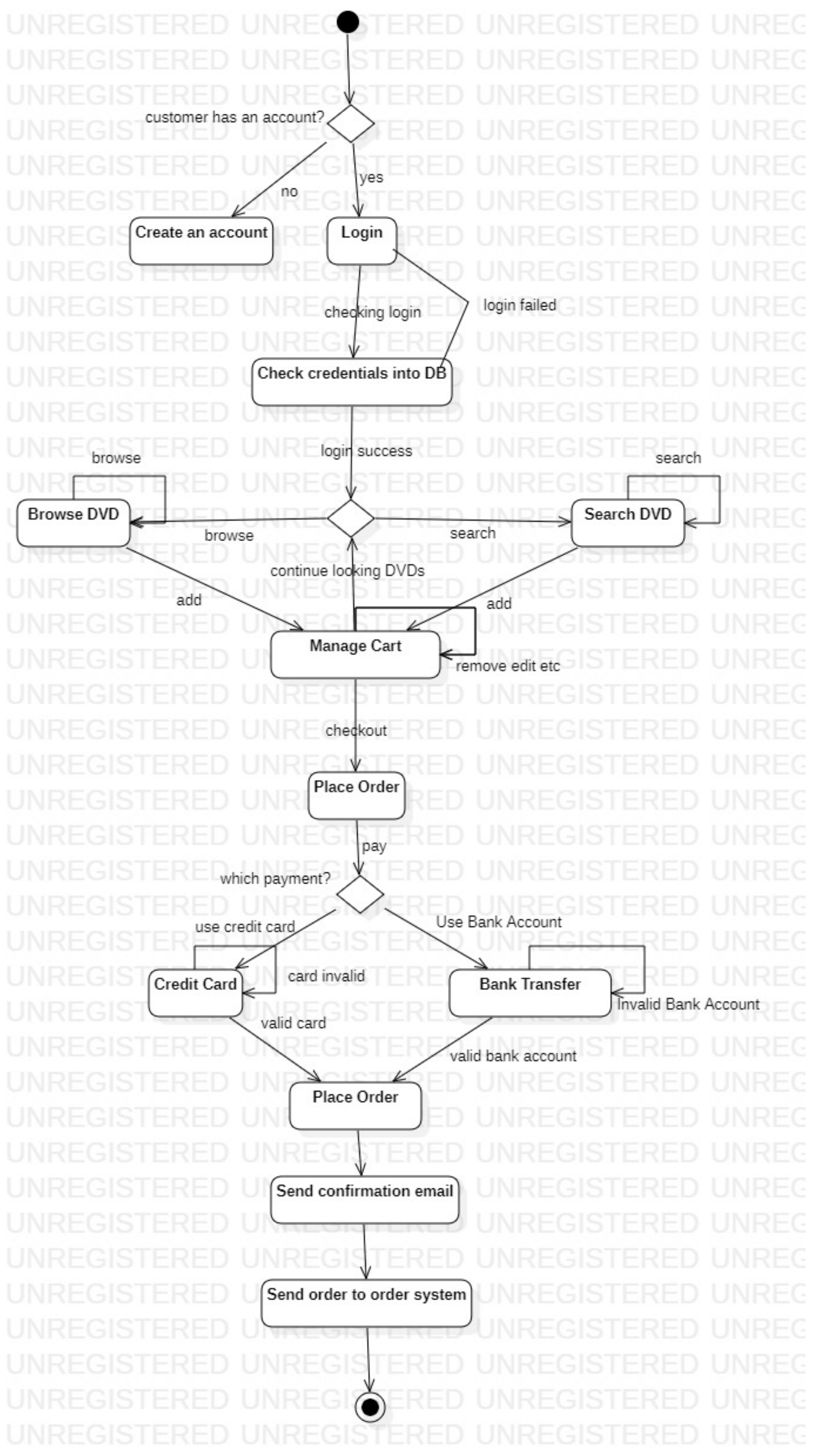
Activity Diagrams:

I have decided to leave the original diagrams in this case because I think that the original approach, I was planning was very good and despite of not being able to implement them completely, I think that it is worth it to have them here.

Activity Diagram for the Subscription scenario: I would have loved to implement the activity diagram above but, unfortunately, I did not have enough time for it.

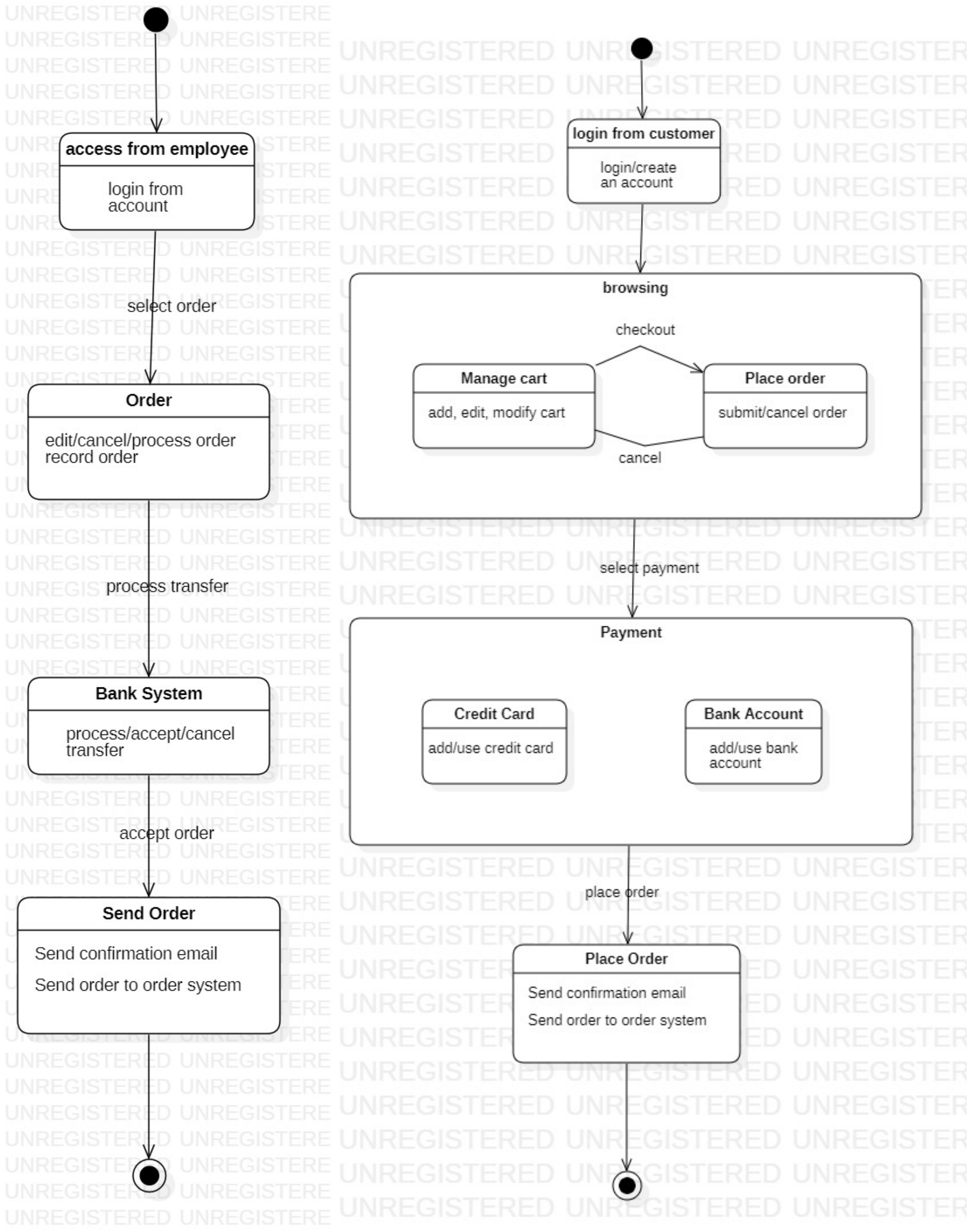


Activity diagram for the order a DVD scenario: For this diagram I have been able to implement most of the features shown with exception of the create an account and bank transfer scenarios.



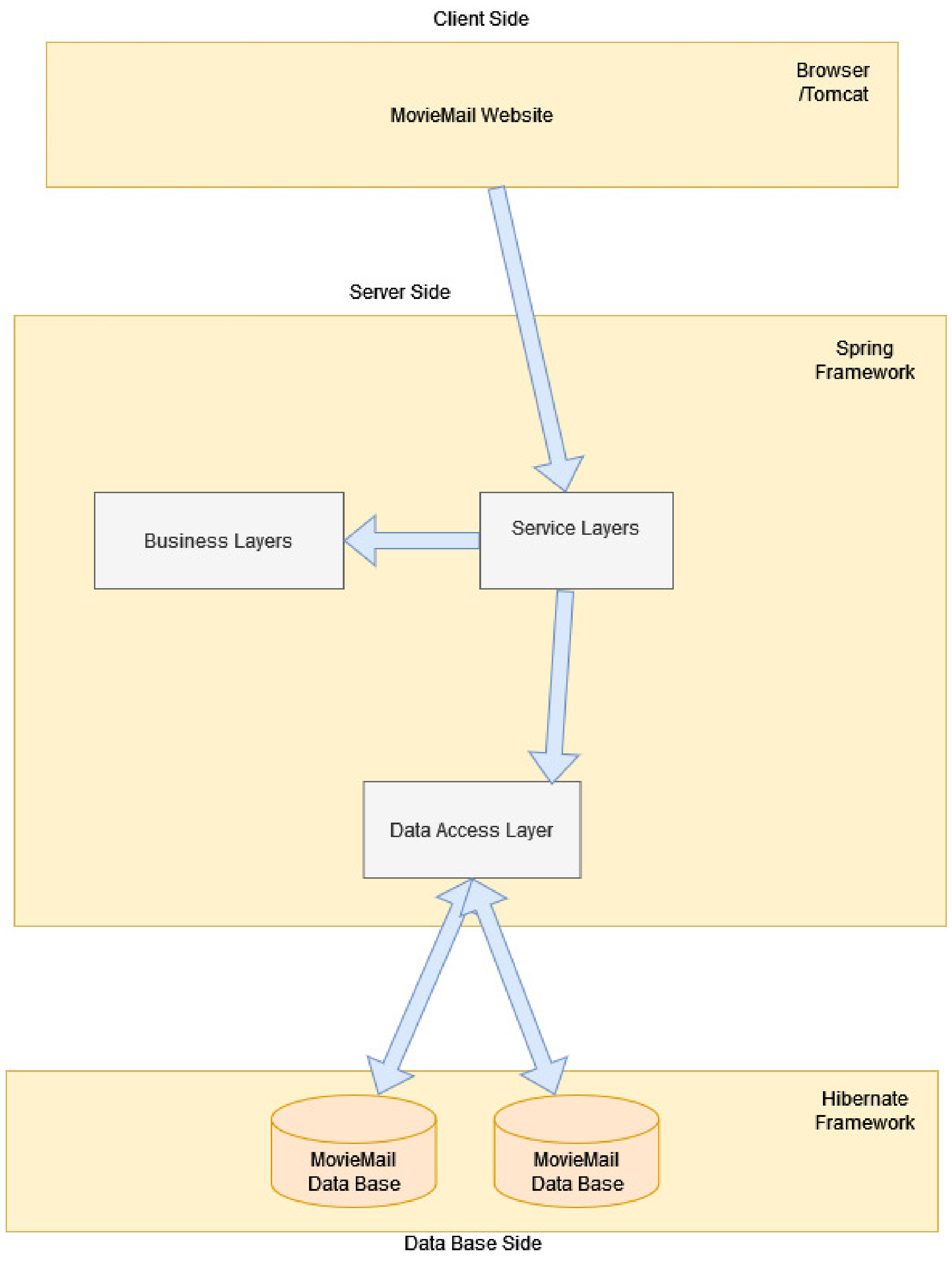
State Machine Diagram:

Here the situation is the same. In the employee state machine diagram, I was not able to implement finally but as before I consider that it is worth it to have it here in my project.



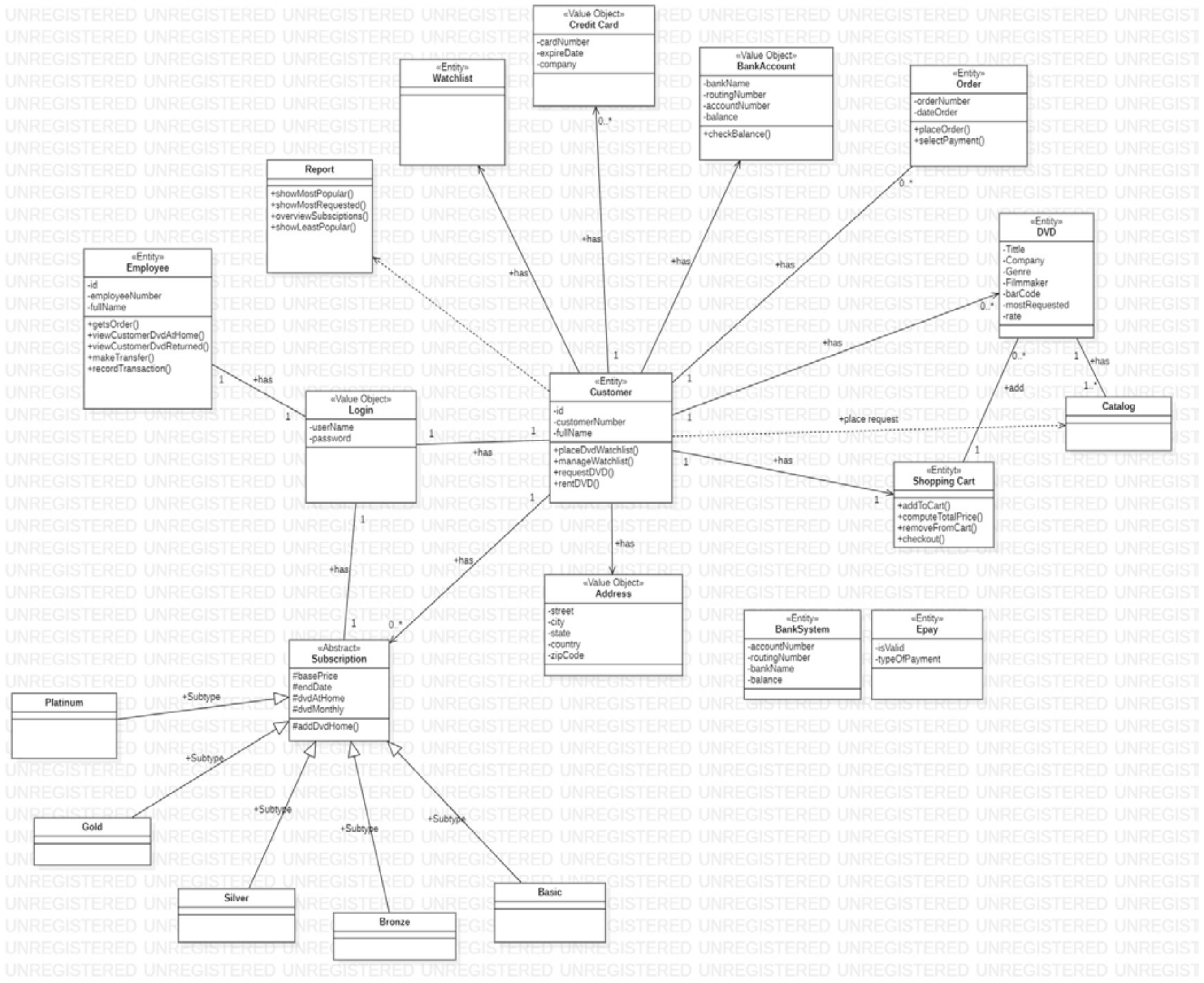
Architecture Diagram:

After finishing the project, I realized that the Architecture diagram I made originally is not the same as how I have implemented my project, therefore, I have made some changes. In my projects both Web MVC and RESTFull API, my service and controller classes communicate together but the controller classes never access the data base directly. It is always through the Service Classes. So, the Architecture Diagram I leave here is the appropriate it for my project.



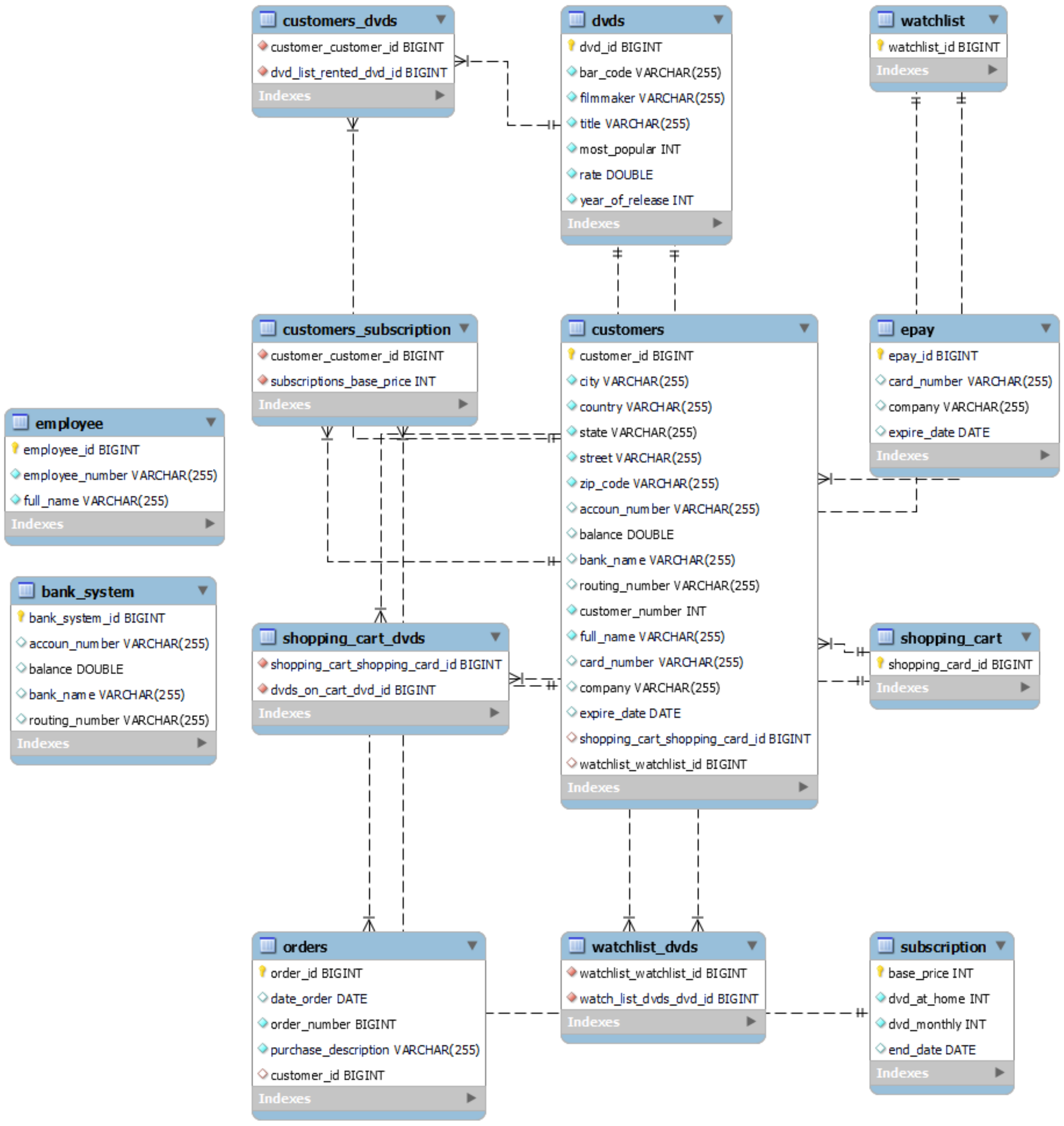
Class Diagram:

Like we discussed at the beginning of my project I modified completely my class diagram to start being able to implement my project. This class diagram is the reflection of the classes and business logic I have in my project. Some of the methods are no implemented because I did not have enough time to do it.



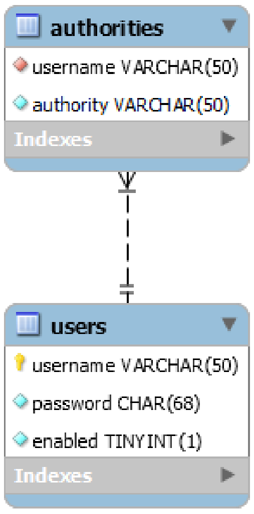
ER – Movie Mail Data Base:

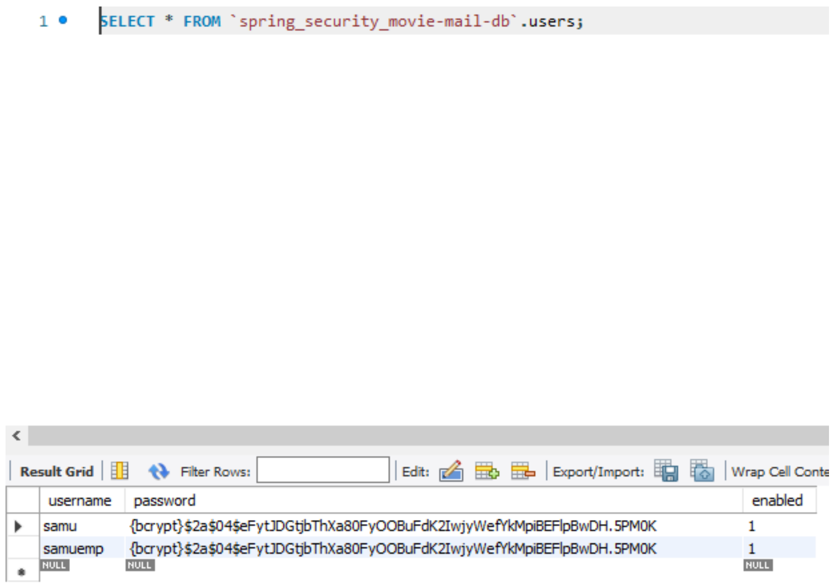
Here is the diagram of the database and its relationships between entities from the RESTFull API project



ER Movie Mail Login Data Base:

In this Data base is store the information of the customers and employees of the application. The password is encrypted with the **bcrypt** technology as the capture of the data base shows.





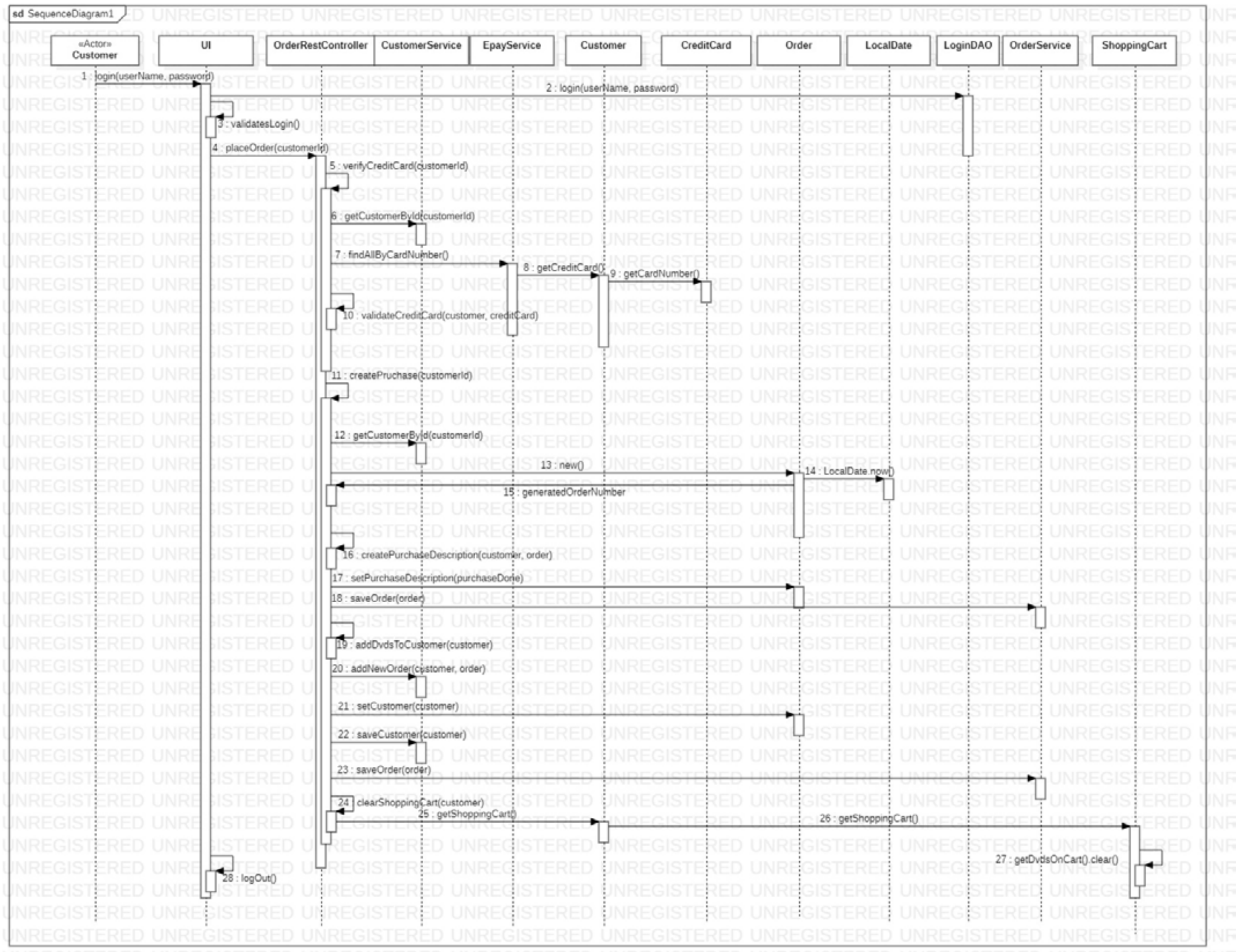
```
SELECT * FROM `spring_security_movie-mail-db`.users;
```

username	password	enabled
samu	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
samuemp	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
NULL	NULL	NULL

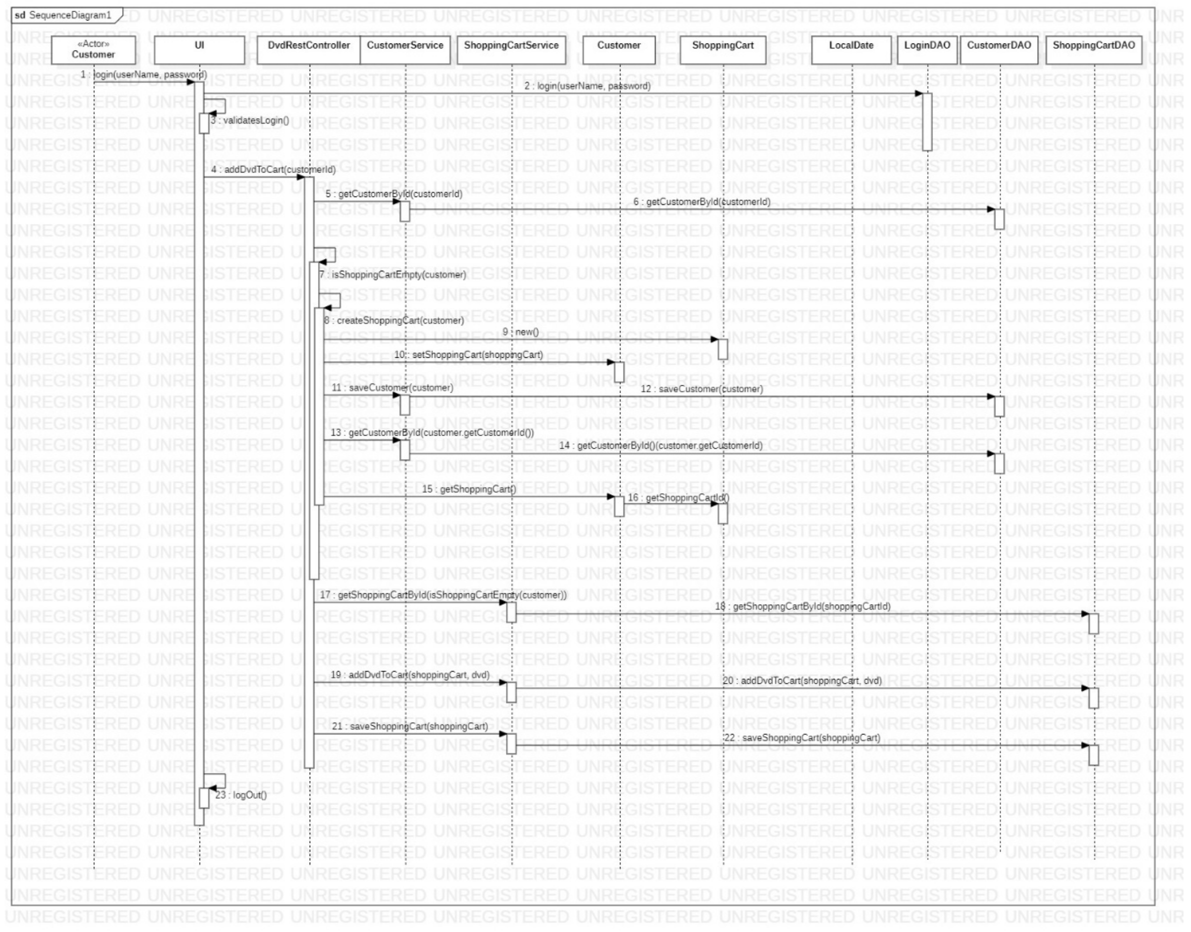
Sequence Diagrams:

Here are the new sequence diagrams I made today for two of the main features I made for this project. One shows how to make an order for a DVD and the other one shows the sequence of adding a DVD in the Shopping Cart.

Place Order for a DVD: This diagram is simplified because it doesn't fit the whole process and methods to show it here as a presentation like the ServiceDAOs classes, therefore, the most important ones are reflected here.


sd SequenceDiagram1
«Actor» Customer
UI
OrderRestController
CustomerService
EpayService
Customer
CreditCard
Order
LocalDate
LoginDAO
OrderService
ShoppingCart
1 : login(userName, password)
2 : login(userName, password)
3 : validatesLogin()
4 : placeOrder(customerId)
5 : verifyCreditCard(customerId)
6 : getCustomerById(customerId)
7 : findAllByCardNumber()
8 : getCreditCard()
9 : getCardNumber()
10 : validateCreditCard(customer, creditCard)
11 : createPruchase(customerId)
12 : getCustomerById(customerId)
13 : new()
14 : LocalDate.now()
15 : generatedOrderNumber
16 : createPurchaseDescription(customer, order)
17 : setPurchaseDescription(purchaseDone)
18 : saveOrder(order)
19 : addDvdsToCustomer(customer)
20 : addNewOrder(customer, order)
21 : setCustomer(customer)
22 : saveCustomer(customer)
23 : saveOrder(order)
24 : clearShoppingCart(customer)
25 : getShoppingCart()
26 : getShoppingCart()
27 : getDvdsOnCart().clear()
28 : logOut()

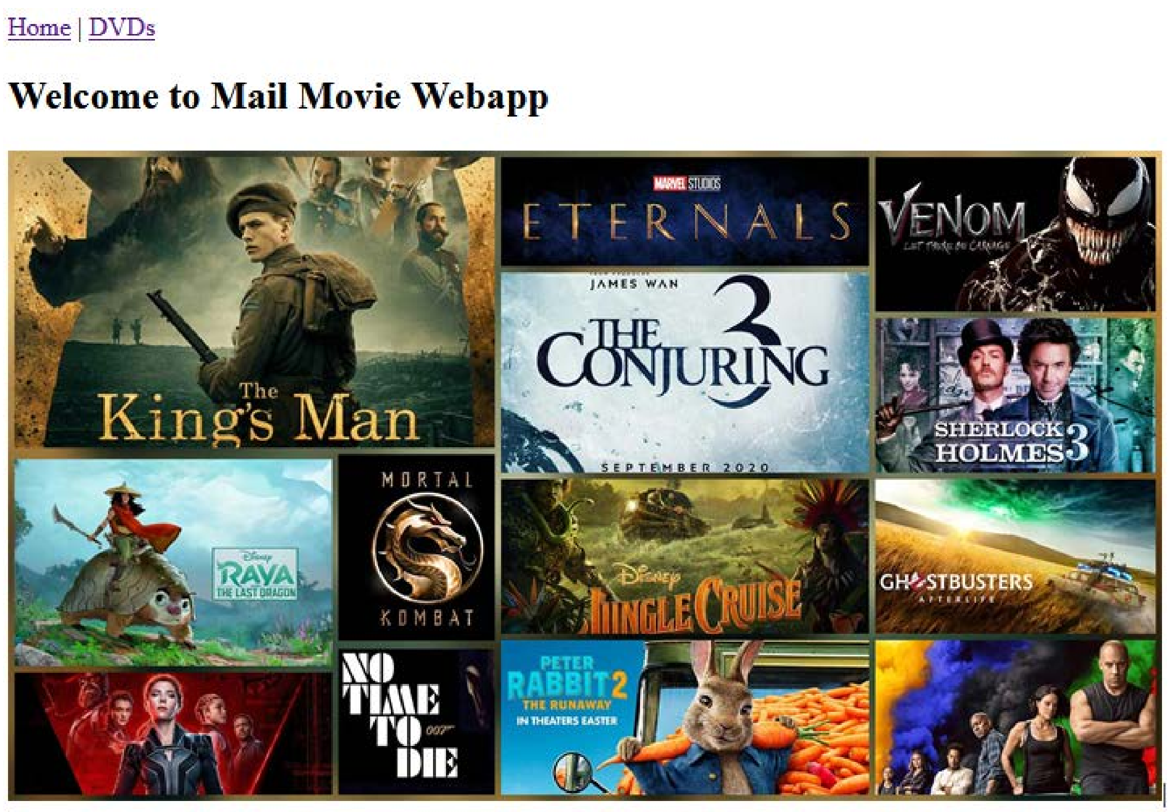
Add DVD to Cart Diagram:


sd SequenceDiagram1
«Actor» Customer
UI
DvdRestController
CustomerService
ShoppingCartService
Customer
ShoppingCart
LocalDate
LoginDAO
CustomerDAO
ShoppingCartDAO
1 : login(userName, password)
2 : login(userName, password)
3 : validatesLogin()
4 : addDvdToCart(customerId)
5 : getCustomerById(customerId)
6 : getCustomerById(customerId)
7 : isShoppingCartEmpty(customer)
8 : createShoppingCart(customer)
9 : new()
10 : setShoppingCart(shoppingCart)
11 : saveCustomer(customer)
12 : saveCustomer(customer)
13 : getCustomerById(customer.getCustomerId())
14 : getCustomerById()(customer.getCustomerId)
15 : getShoppingCart()
16 : getShoppingCartId()
17 : getShoppingCartById(isShoppingCartEmpty(customer))
18 : getShoppingCartById(shoppingCartId)
19 : addDvdToCart(shoppingCart, dvd)
20 : addDvdToCart(shoppingCart, dvd)
21 : saveShoppingCart(shoppingCart)
22 : saveShoppingCart(shoppingCart)
23 : logOut()

User Interface:

Here I leave the captures of the main features of my project from the User Interface presentation:

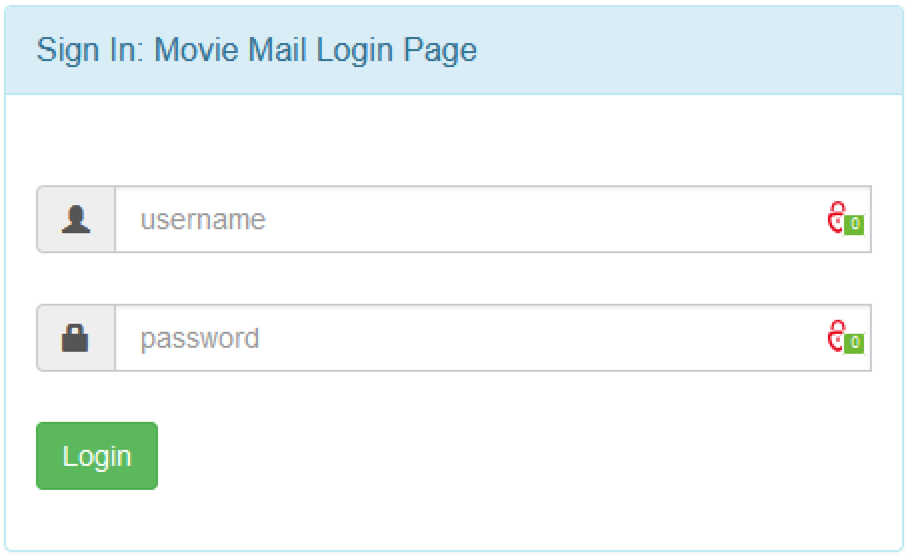
Home Page:



Home | DVDs

Welcome to Mail Movie Webapp

Login page:



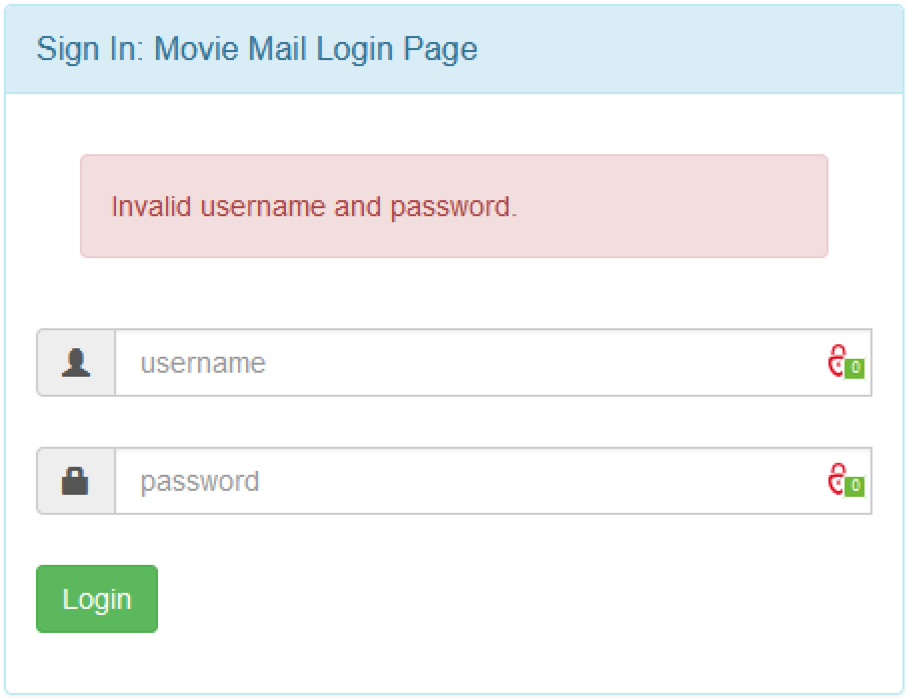
Sign In: Movie Mail Login Page

username

password

Login

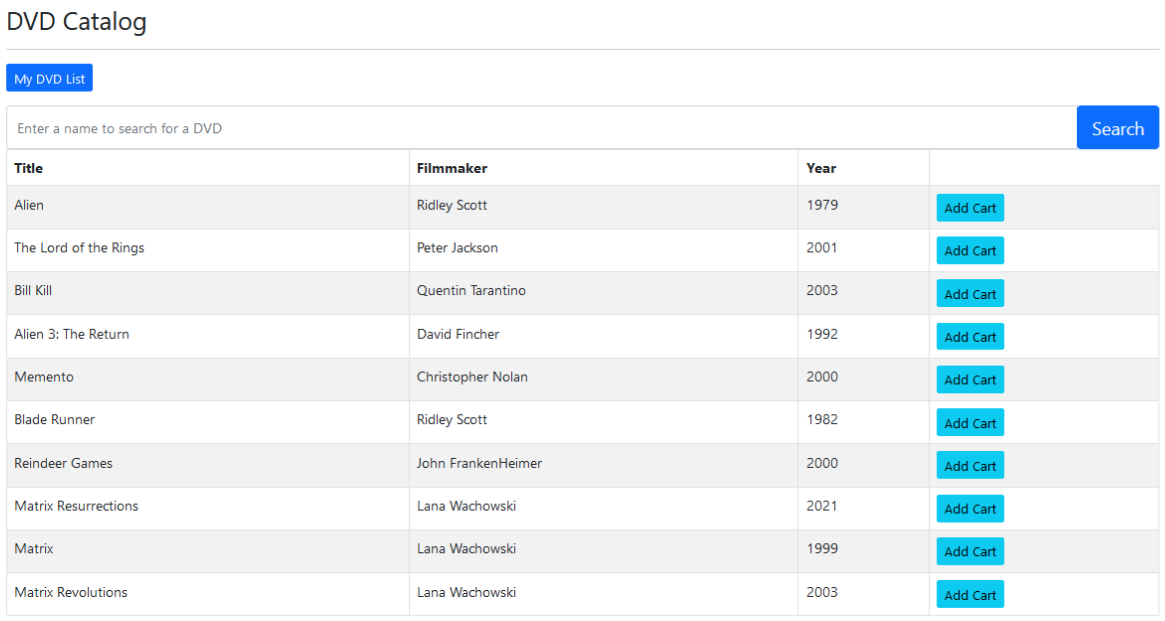
Login page error:



Sign In: Movie Mail Login Page

Invalid username and password.

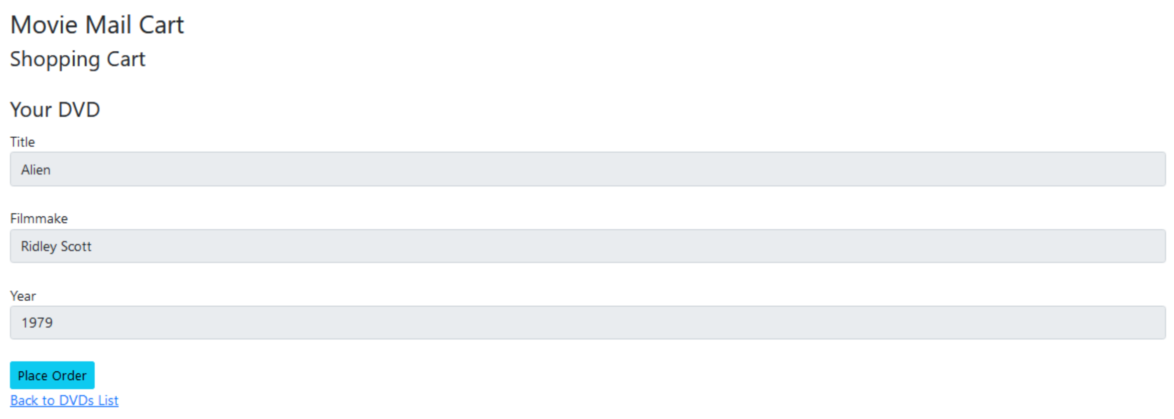
DVD Catalog:



DVD Catalog

Title	Filmmaker	Year	
Alien	Ridley Scott	1979	Add Cart
The Lord of the Rings	Peter Jackson	2001	Add Cart
Bill Kill	Quentin Tarantino	2003	Add Cart
Alien 3: The Return	David Fincher	1992	Add Cart
Memento	Christopher Nolan	2000	Add Cart
Blade Runner	Ridley Scott	1982	Add Cart
Reindeer Games	John FrankenHeimer	2000	Add Cart
Matrix Resurrections	Lana Wachowski	2021	Add Cart
Matrix	Lana Wachowski	1999	Add Cart
Matrix Revolutions	Lana Wachowski	2003	Add Cart

Place Order for DVD Page:



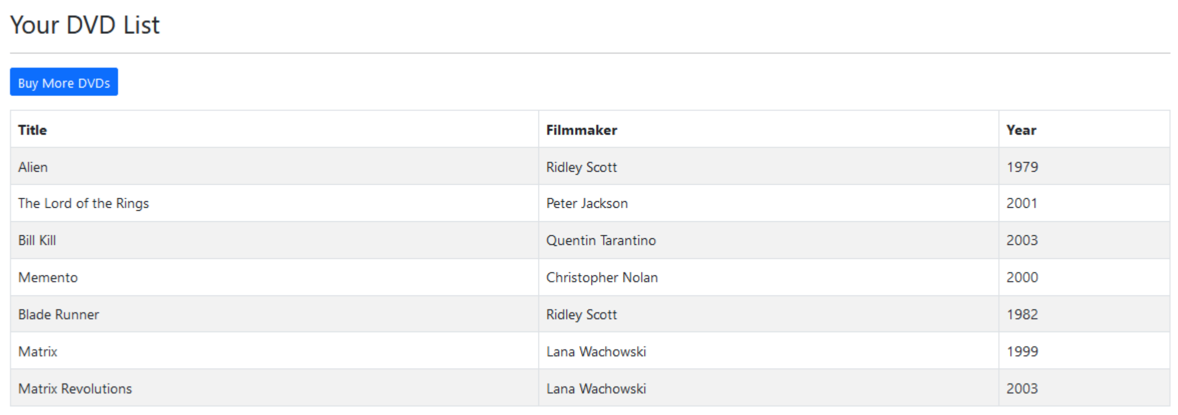
Movie Mail Cart

Shopping Cart

Your DVD

Title	Alien
Filmmake	Ridley Scott
Year	1979

Customer DVD List:



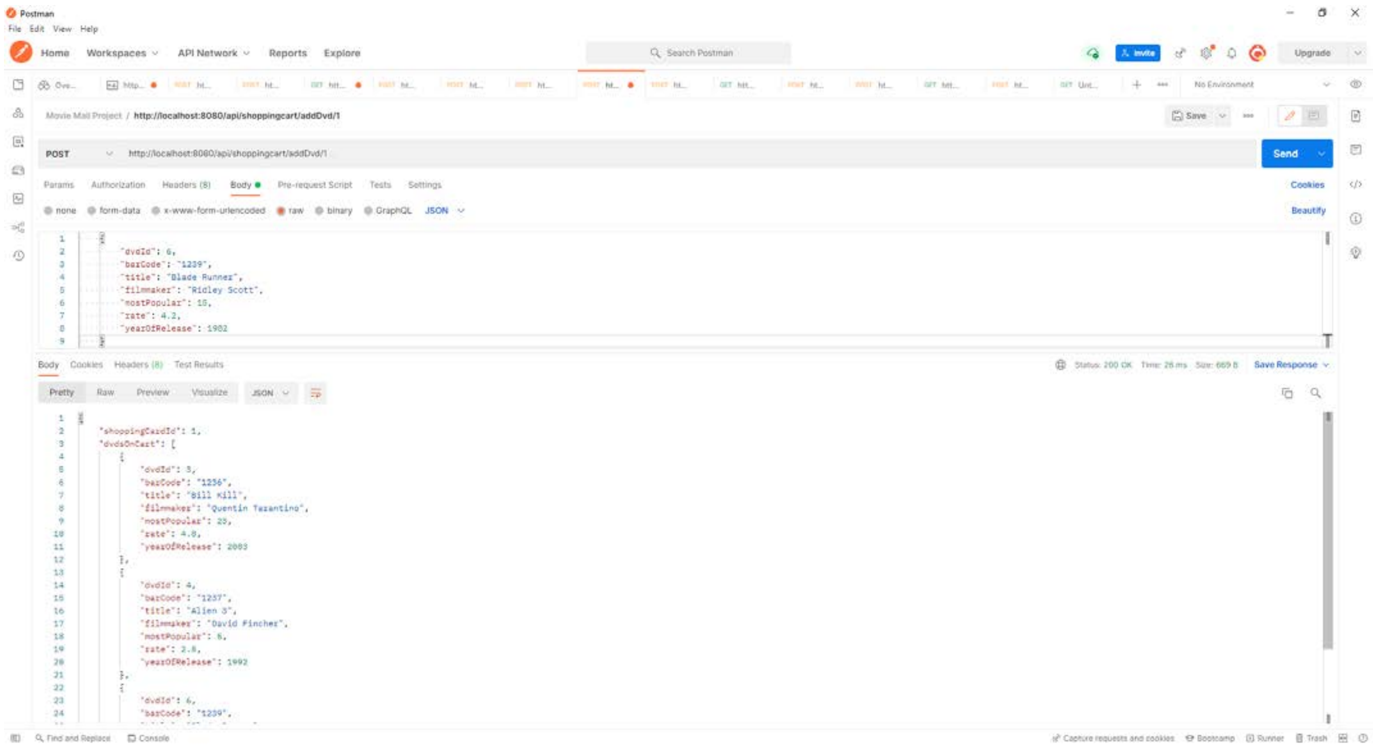
Your DVD List

Title	Filmmaker	Year
Alien	Ridley Scott	1979
The Lord of the Rings	Peter Jackson	2001
Bill Kill	Quentin Tarantino	2003
Memento	Christopher Nolan	2000
Blade Runner	Ridley Scott	1982
Matrix	Lana Wachowski	1999
Matrix Revolutions	Lana Wachowski	2003

END Points RESTFull API Project:

Here I show some of the most important end points of this project.

Add DVD to Cart end point:

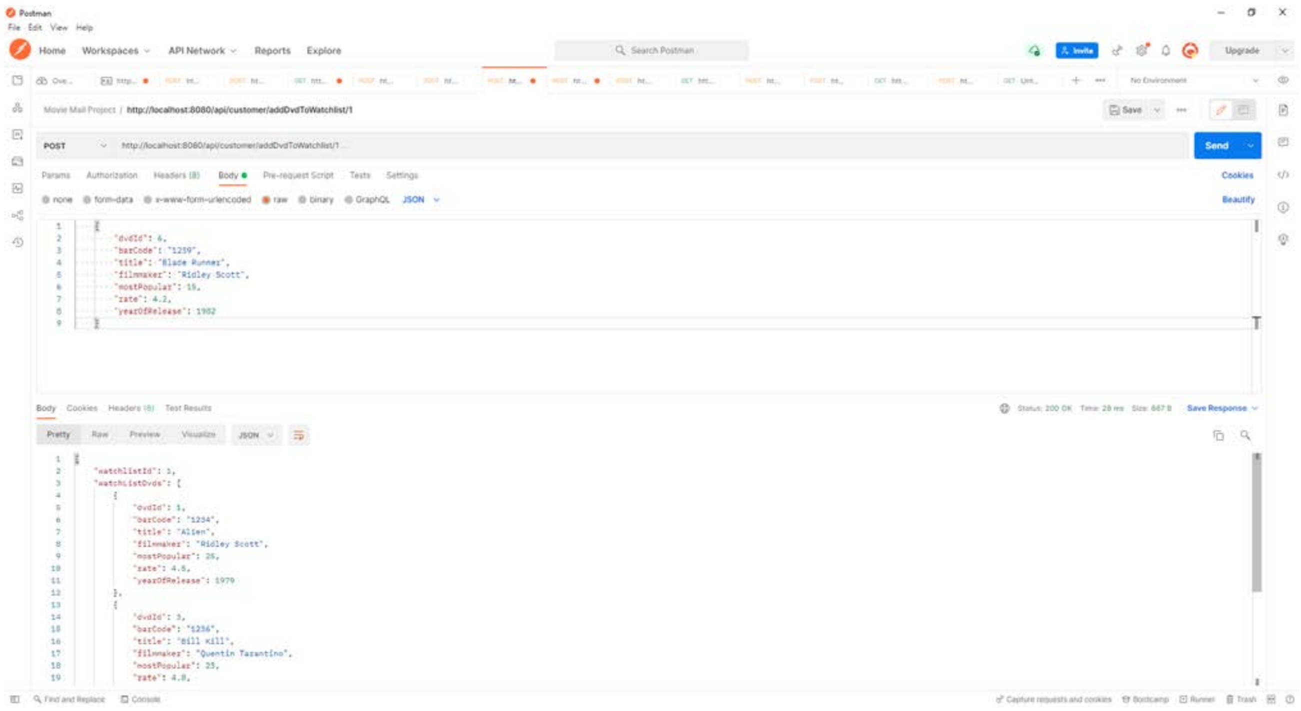


```
POST http://localhost:8080/api/shoppingcart/addDvd/1
```

```
{
    "dvdId": 6,
    "barCode": "1239",
    "title": "Blade Runner",
    "filmmaker": "Ridley Scott",
    "mostPopular": 15,
    "rate": 4.2,
    "yearOfRelease": 1982
}
```

```
{
    "shoppingCardId": 1,
    "dvdsOnCart": [
        {
            "dvdId": 3,
            "barCode": "1236",
            "title": "Bill Kill",
            "filmmaker": "Quentin Tarantino",
            "mostPopular": 25,
            "rate": 4.8,
            "yearOfRelease": 2003
        },
        {
            "dvdId": 4,
            "barCode": "1237",
            "title": "Alien 3",
            "filmmaker": "David Fincher",
            "mostPopular": 6,
            "rate": 2.8,
            "yearOfRelease": 1992
        },
        {
            "dvdId": 6,
            "barCode": "1239",
```

Add DVD to Watchlist end point:

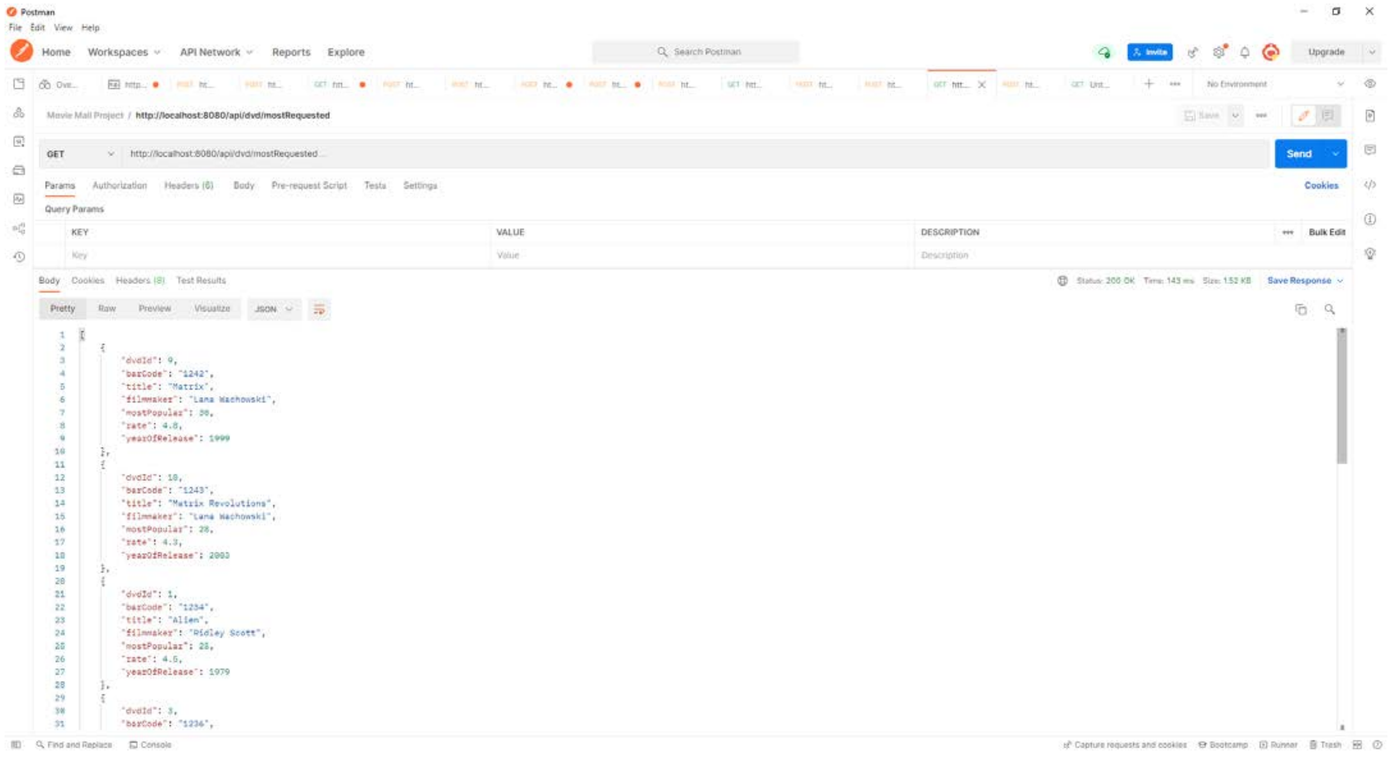


```
POST http://localhost:8080/api/customer/addDvdToWatchlist/1
```

```
{
    "dvdId": 6,
    "barCode": "1239",
    "title": "Blade Runner",
    "filmmaker": "Ridley Scott",
    "mostPopular": 15,
    "rate": 4.2,
    "yearOfRelease": 1982
}
```

```
{
    "watchlistId": 1,
    "watchListDvds": [
        {
            "dvdId": 1,
            "barCode": "1234",
            "title": "Alien",
            "filmmaker": "Ridley Scott",
            "mostPopular": 26,
            "rate": 4.6,
            "yearOfRelease": 1979
        },
        {
            "dvdId": 3,
            "barCode": "1236",
            "title": "Bill Kill",
            "filmmaker": "Quentin Tarantino",
            "mostPopular": 25,
            "rate": 4.8,
```

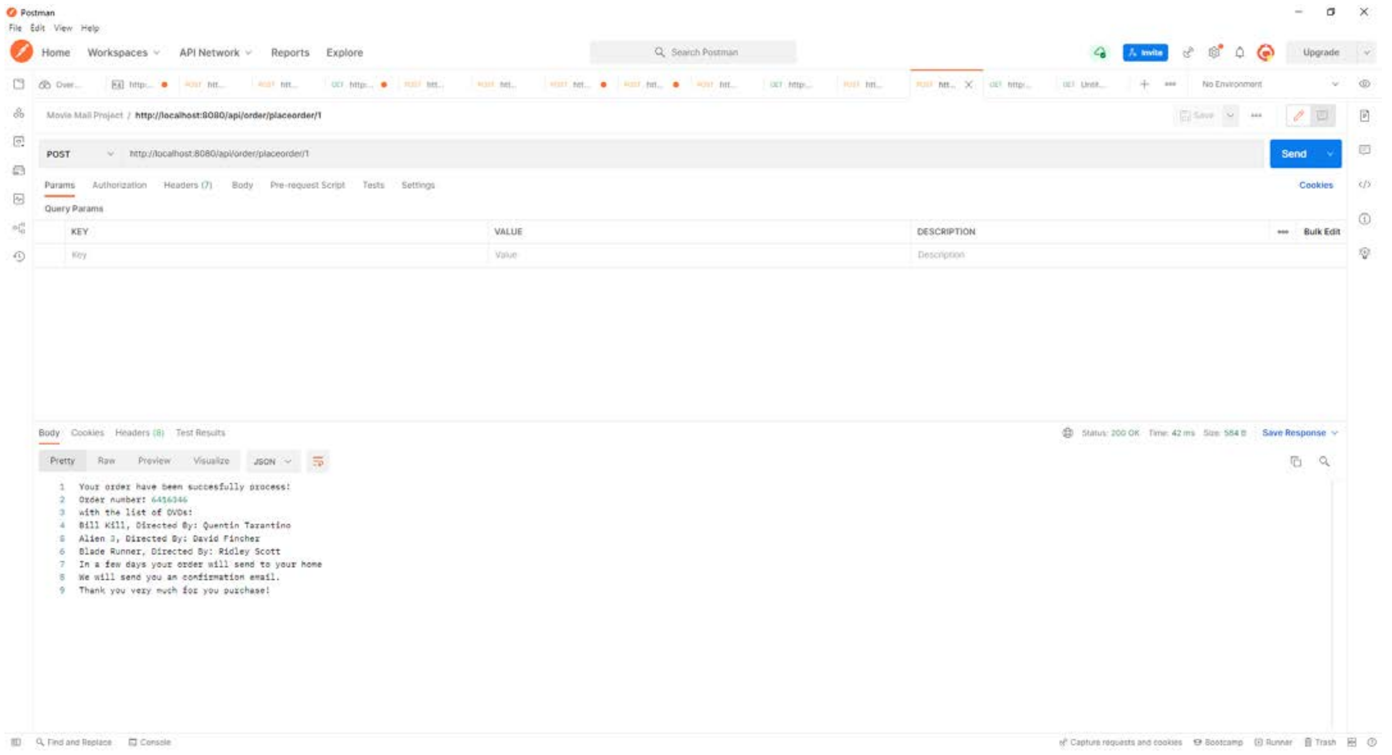
Show List Most Requested:


GET http://localhost:8080/api/dvd/mostRequested

Status: 200 OK Time: 143 ms Size: 1.52 KB

```
[
    {
        "dvdId": 9,
        "barCode": "1242",
        "title": "Matrix",
        "filmmaker": "Lana Wachowski",
        "mostPopular": 30,
        "rate": 4.8,
        "yearOfRelease": 1999
    },
    {
        "dvdId": 10,
        "barCode": "1243",
        "title": "Matrix Revolutions",
        "filmmaker": "Lana Wachowski",
        "mostPopular": 28,
        "rate": 4.3,
        "yearOfRelease": 2003
    },
    {
        "dvdId": 1,
        "barCode": "1234",
        "title": "Alien",
        "filmmaker": "Ridley Scott",
        "mostPopular": 28,
        "rate": 4.5,
        "yearOfRelease": 1979
    },
    {
        "dvdId": 3,
        "barCode": "1236",
```

Place Order end point:


POST http://localhost:8080/api/order/placeorder/1

Status: 200 OK Time: 42 ms Size: 584 B

```
Your order have been succesfully process!
Order number: 6416346
with the list of DVDs:
Bill Kill, Directed By: Quentin Tarantino
Alien 3, Directed By: David Fincher
Blade Runner, Directed By: Ridley Scott
In a few days your order will send to your home
We will send you an confirmation email.
Thank you very much for you purchase!
```

Conclusion:

I have tried to apply for this project all the knowledge we haver learn from the beginning of course with the Agile methodologies to the last lessons for the course related to Spring boot technologies.

However, I am aware that I could finish the project properly, leaving several features without being implemented as the employee and subscription features. I am aware as well that I could not refined the implementation I have done, improving my code, cleaning of fixit possible bugs as improper as trying to place an order of an DVD already rented for the customer etc.

I have done everything I could for the space of time given for this project.

Lastly, I would like to thank you professor for all you help during this project. I would not have finished it ever without you. Your help fixing errors in my project allowed me to present a solid project today. Thank you very much for that.

Thank you for everything and I hope this project has been enough for this course. I have done my best. I hope you feel happy with my work.

All the best professor!!!